08

Configuración de seguridad #1

Transcripción

[00:00] Listo. Ya tengo casi todo para decirle a Spring que yo quiero que mi proceso de autenticación sea stateless y no stateful. Ahora solo falta una cosa y es una configuración en específico. Como les dije, en este caso las configuraciones que estamos realizando no son configuraciones digamos a nivel de archivos properties en el cual yo le puedo decir true o false, si quiero cambiar el comportamiento de Spring.

[00:27] En este caso es implementar interfaces propias de Spring como estamos haciendo acá, para yo personalizar el comportamiento por defecto que Spring Security tiene. Entonces, para esto yo en mi paquete de Security que he creado, voy a crear una nueva clase para mis configuraciones y aquí voy a seguir SecurityConfigurations.

[00:58] No lo agregó aun. Y como toda configuración que yo quiero que sea escaneada por Spring, yo le voy a marcar como @Configuration que está aquí.

[01:10] Entonces Spring con esta anotación al igual que aquí con service, Spring va a venir aquí y va a decir: “Esta es una configuración, entonces la voy a escanear, porque en el orden de creación de los objetos de Spring, primero se escanean los que están anotados con @Configuration porque se sobreentiende que son prerrequisitos para que la aplicación para que otros objetos de la aplicación puedan ser creados”. Esto es solo un dato curioso.

[01:37] Bueno, ¿qué más necesitamos? Necesitamos un tipo de objeto que me retorne aquí para que implemente una autenticación stateless aquí en Spring

¿A qué me refiero con un tipo de objeto? Lo que necesito es un tipo de objeto public, tengo que declarar este método, que va a ser mi securyFilterChain, de aquí de Springframe, esto de aquí. Nombre, el mismo y ya está.

[02:12] Esto me tiene que retornar un objeto del tipo SecurityFilterChain. Como parámetro lo que él va a recibir es un objeto del tipo httpSecurity, vamos a dejarlo con httpSecurity. Para que Spring sepa que esta es una configuración del contexto de seguridad, yo aquí lo que tengo que decirle es @EnableWebSecurity.

[02:41] Entonces con estas dos configuraciones yo le digo a Spring primero que todo: esta es una clase de configuración; segundo: habilítame módulo WebSecurity para esta clase de configuración. ¿Por qué? Porque con el @EnableWebSecurity yo le voy a decir que este método de aquí está siendo implementado para sobrescribir el comportamiento de autenticación que yo quiero. ¿Qué necesito ahora?

[03:06] Segundo, voy a retornar mi httpSecurity.csrf(). Vemos que tenga la terminación aquí del csrf. ¿Qué significa esto? El Cross-site request forgery y método de protection. Esto es para evitar suplantación de identidad, pero como no estoy usando autenticación stateful no lo necesito por ahora, no se preocupen. Entonces era solo para mostrarles dónde está este tipo de seguridad que Spring provee, está implementando seguridad de una aplicación web.

[03:41] Llamen este método de aquí porque es el que les va a proporcionar ese tipo de protección. Entonces yo no lo voy a llamar por ahora, entonces lo que voy a decirle aquí es que lo quiero deshabilitado, porque como ya tengo autenticación vía token o mi autenticación y autorización va ser vía web token no tengo por qué tener una protección de Cross-site request forgery, no tiene sentido.

[04:11] La autenticación stateless ya me protege contra ese tipo de ataques. Entonces no hay problema. ¿Qué más necesito aquí? Aquí lo que necesito es decirle con sesión, sessionManagement.